

NAME: Hetvi Patel

CSC 138

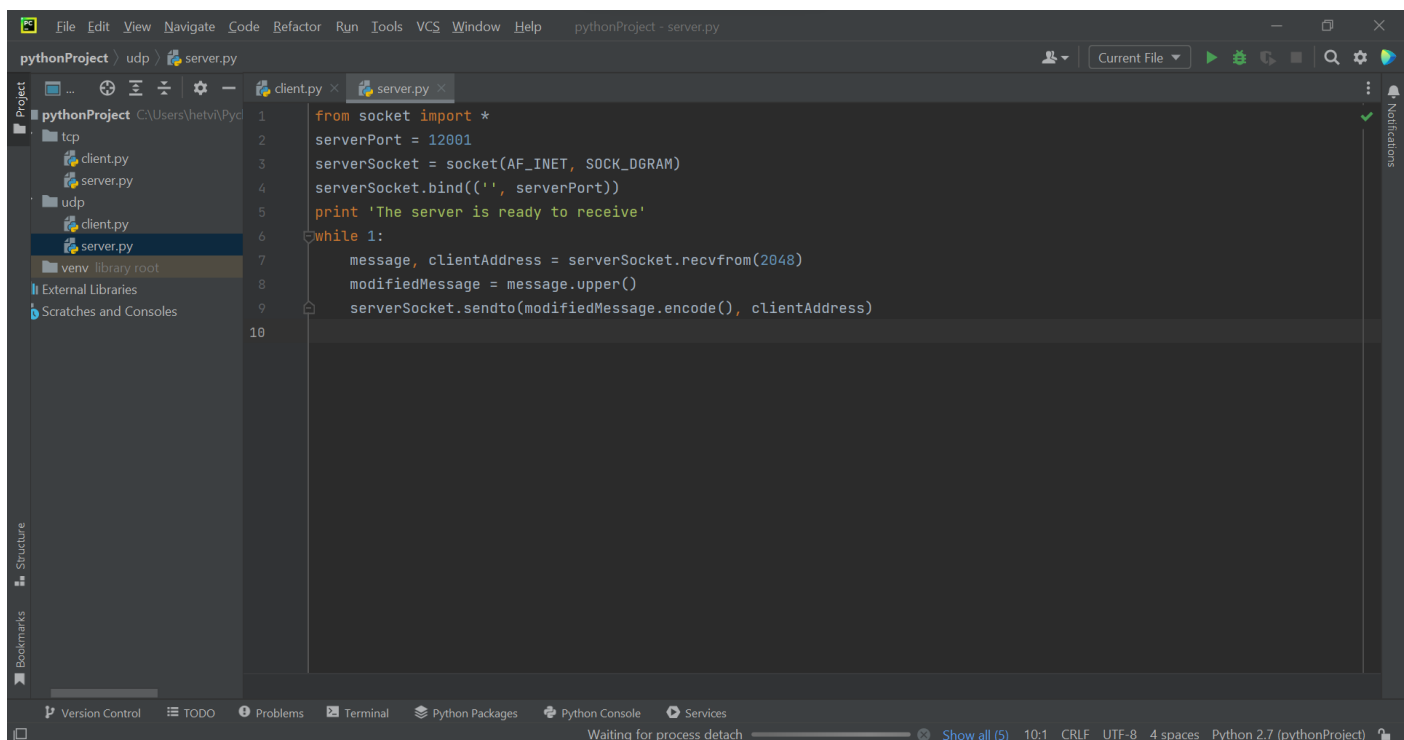
DATE: 10/05/2022

Socket 1

UDP Screenshots :

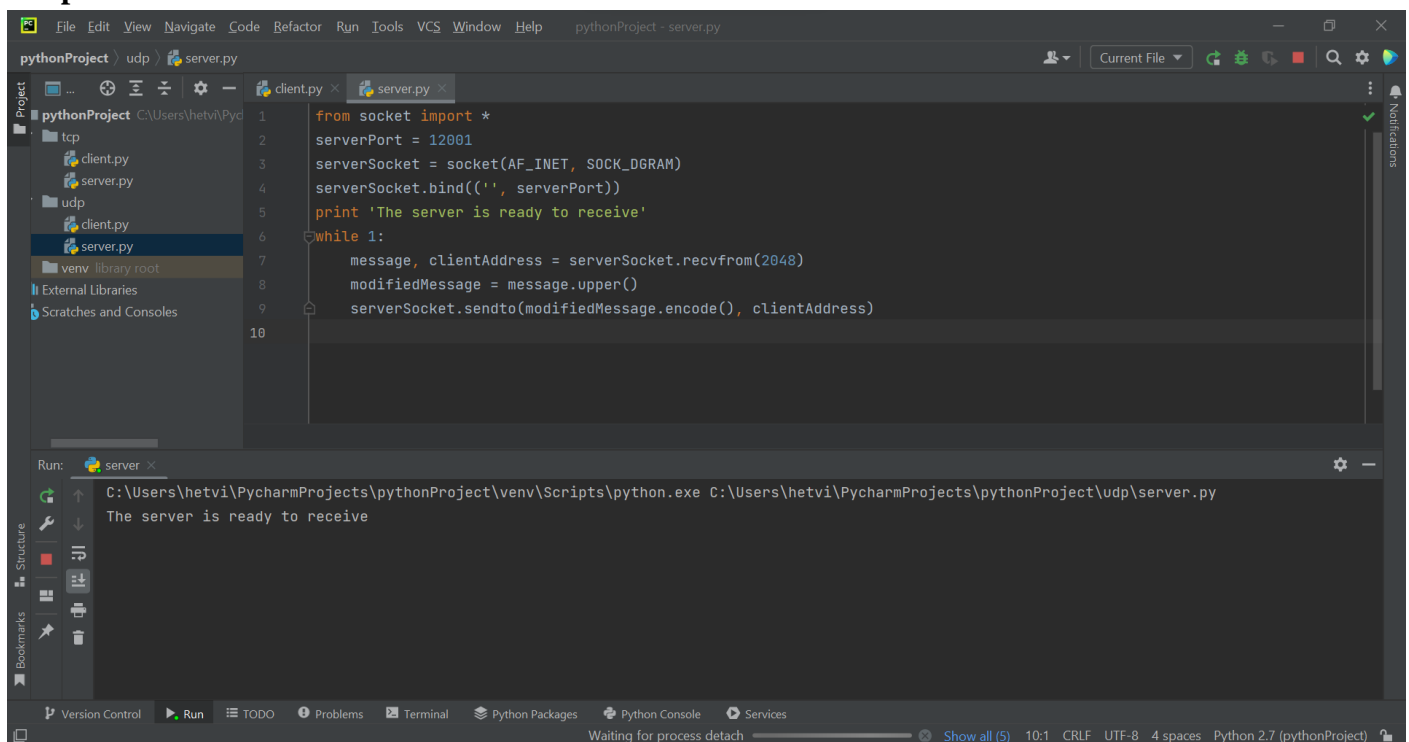
Source code for server:

```
from socket import *
serverPort = 30002
serverSocket = socket(AF_INET, SOCK_DGRAM)
serverSocket.bind('', serverPort)
print('The server is ready to receive')
while 1:
    message, clientAddress = serverSocket.recvfrom(2048)
    modifiedMessage = message.upper()
    serverSocket.sendto(modifiedMessage.encode(), clientAddress)
```



This is the source code for udp server where it prints out ‘The server is ready to receive’. It takes the modifiedMessage and changes it to uppercase letters. The server never requires a close() function.

Output for server:



The screenshot displays the PyCharm IDE interface. The main editor window shows the file `server.py` with the following Python code:

```
1 from socket import *
2 serverPort = 12001
3 serverSocket = socket(AF_INET, SOCK_DGRAM)
4 serverSocket.bind(('', serverPort))
5 print 'The server is ready to receive'
6 while 1:
7     message, clientAddress = serverSocket.recvfrom(2048)
8     modifiedMessage = message.upper()
9     serverSocket.sendto(modifiedMessage.encode(), clientAddress)
10
```

The left sidebar shows the project structure for `pythonProject`, with the `udp` folder and `server.py` file highlighted. The bottom panel shows the Run output for the `server` process, displaying the command executed and the output message:

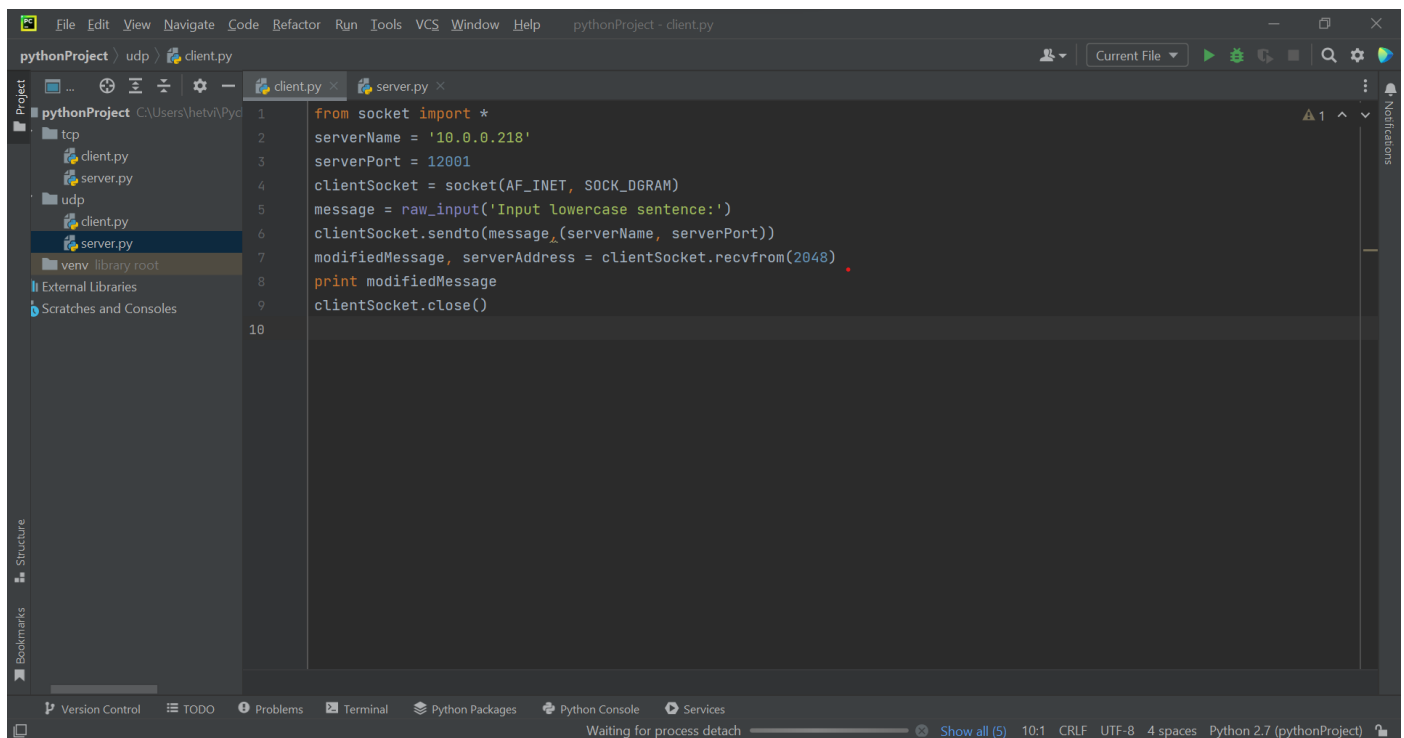
```
Run: server
C:\Users\hetvi\PycharmProjects\pythonProject\venv\Scripts\python.exe C:\Users\hetvi\PycharmProjects\pythonProject\udp\server.py
The server is ready to receive
```

The status bar at the bottom indicates the file encoding is UTF-8, the line length is 10:1, and the Python version is 2.7.

This is the output for udp server where it shows that the server is ready to receive the message from client.

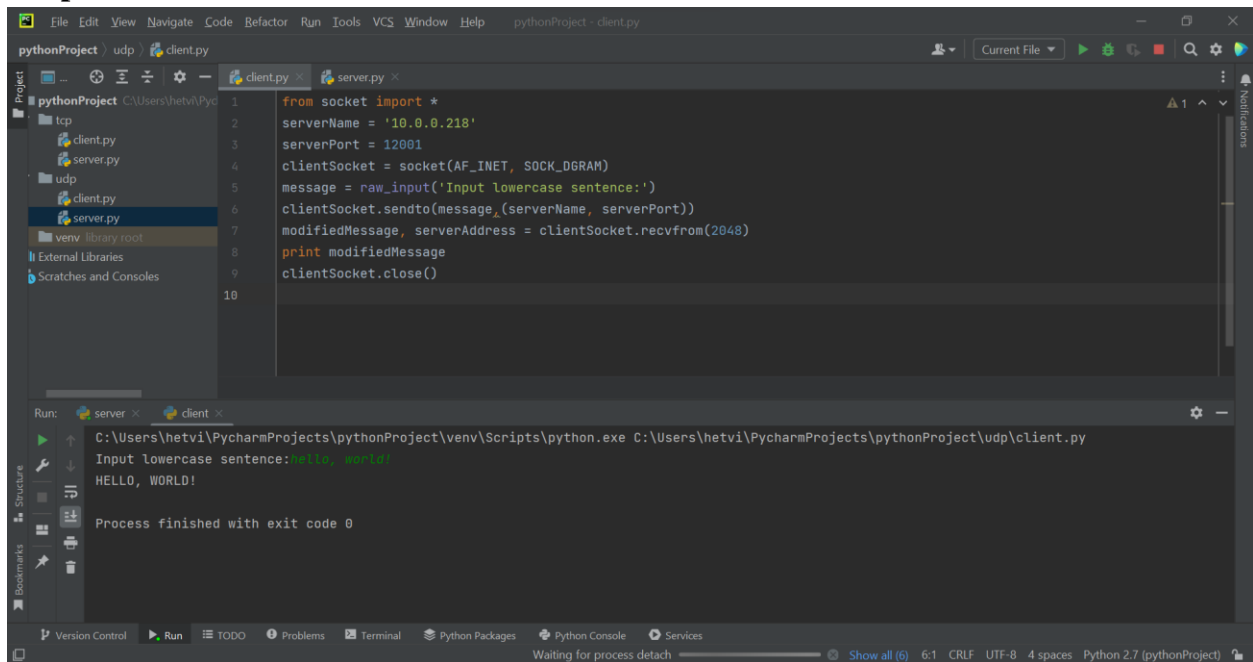
Source code for client:

```
from socket import *
serverName = '10.0.0.218'
serverPort = 30002
clientSocket = socket(AF_INET, SOCK_DGRAM)
message = raw_input('Input lowercase sentence:')
clientSocket.sendto(message, (serverName, serverPort))
modifiedMessage, serverAddress = clientSocket.recvfrom(2048)
print modifiedMessage
clientSocket.close()
```



This is the source code for udp client where it asks to input a lowercase sentence which will then output that sentence as an uppercase sentence using udp server. The udp client file always uses close() function.

Output for client:



```
pythonProject  udp  client.py
pythonProject  client.py  server.py
1  from socket import *
2  serverName = '10.0.0.218'
3  serverPort = 12001
4  clientSocket = socket(AF_INET, SOCK_DGRAM)
5  message = raw_input('Input lowercase sentence:')
6  clientSocket.sendto(message, (serverName, serverPort))
7  modifiedMessage, serverAddress = clientSocket.recvfrom(2048)
8  print modifiedMessage
9  clientSocket.close()
10

Run:  server  client
C:\Users\hetvi\PycharmProjects\pythonProject\venv\Scripts\python.exe C:\Users\hetvi\PycharmProjects\pythonProject\udp\client.py
Input lowercase sentence: hello, world!
HELLO, WORLD!

Process finished with exit code 0
```

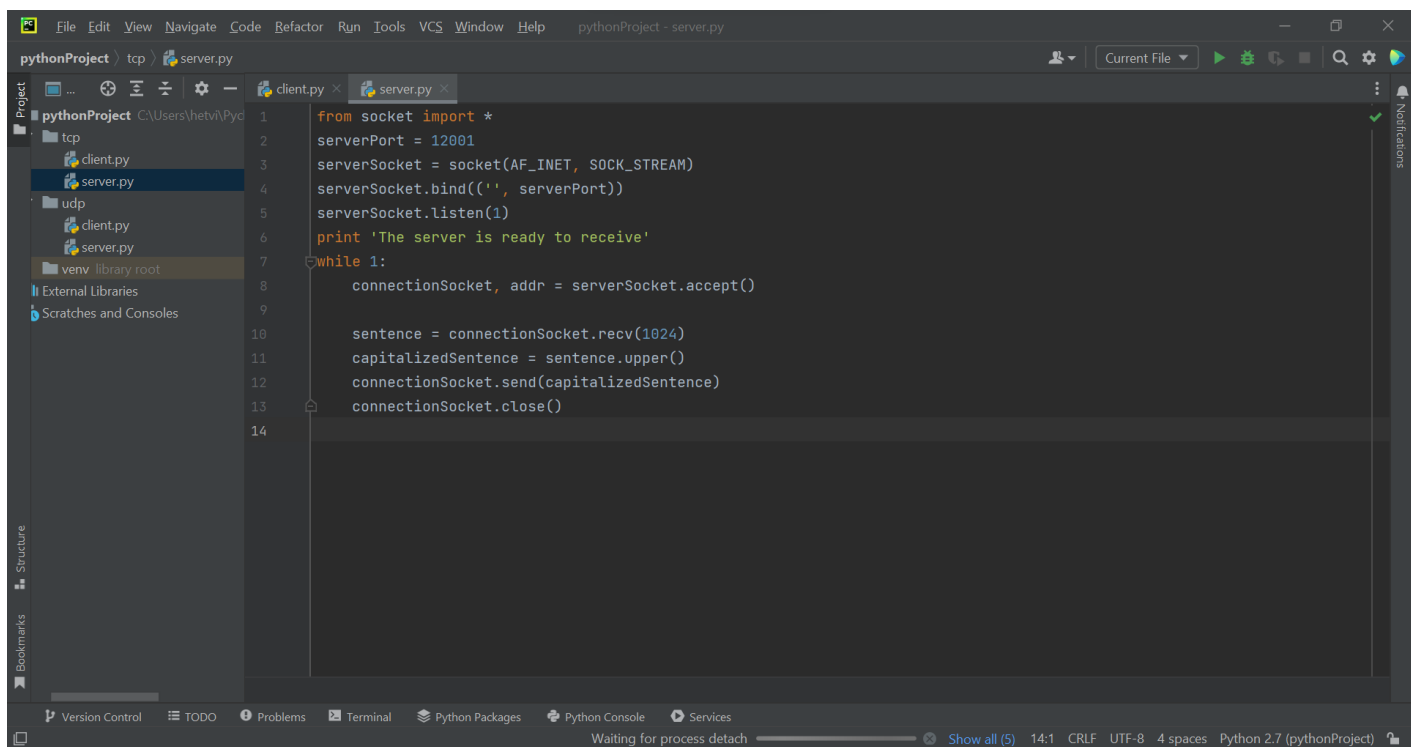
This shows us the output for the lowercase input sentence converted to the uppercase sentence.

TCP Screenshots :

Source code for server:

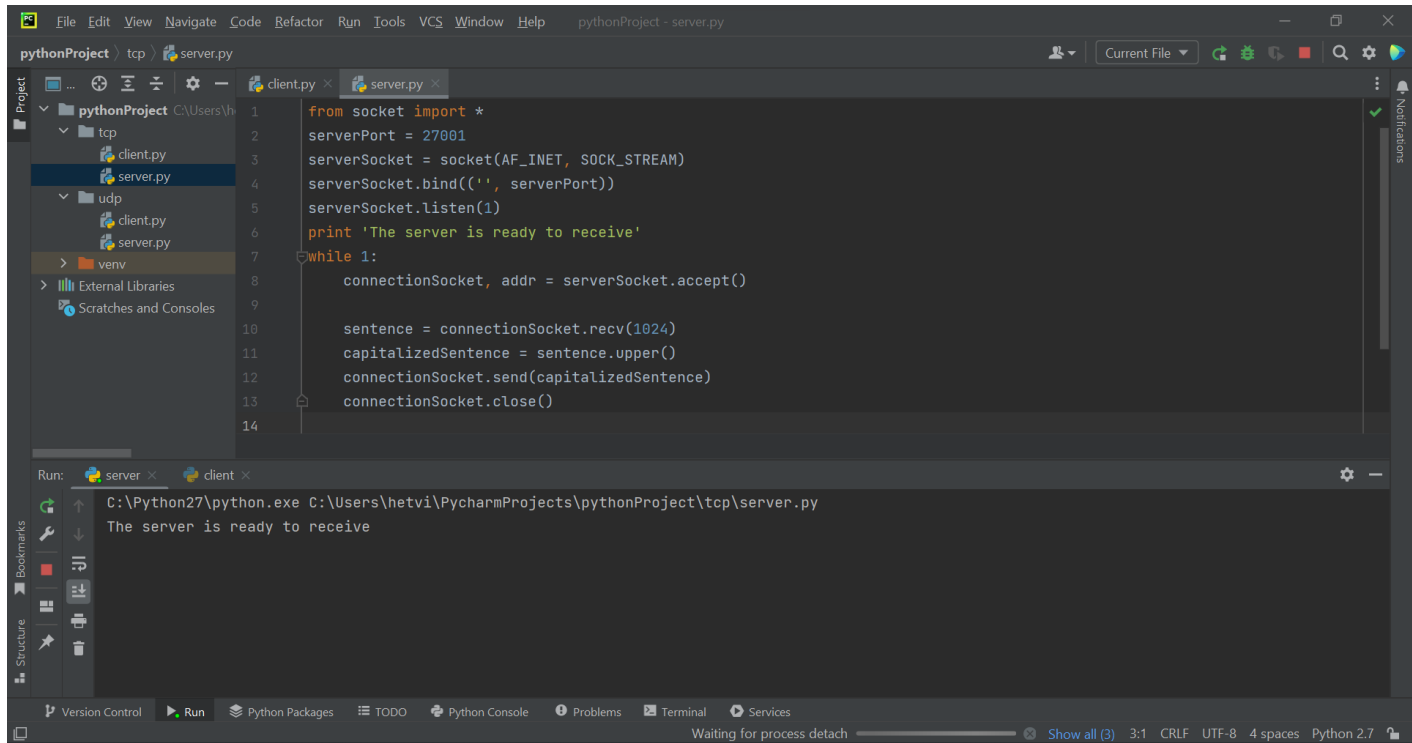
```
from socket import *
serverPort = 27001
serverSocket = socket(AF_INET, SOCK_STREAM)
serverSocket.bind('', serverPort)
serverSocket.listen(1)
print 'The server is ready to receive'
while 1:
    connectionSocket, addr = serverSocket.accept()

    sentence = connectionSocket.recv(1024)
    capitalizedSentence = sentence.upper()
    connectionSocket.send(capitalizedSentence)
    connectionSocket.close()
```



This is the source code for tcp server where it prints out ‘The server is ready to receive’. The server never requires a close() function. Here only connectionSocket is using close() function because it needs to close for server to output.

Output for server:



The screenshot displays the PyCharm IDE interface. The main editor window shows the `server.py` file with the following Python code:

```
1 from socket import *
2 serverPort = 27001
3 serverSocket = socket(AF_INET, SOCK_STREAM)
4 serverSocket.bind(('', serverPort))
5 serverSocket.listen(1)
6 print 'The server is ready to receive'
7 while 1:
8     connectionSocket, addr = serverSocket.accept()
9
10    sentence = connectionSocket.recv(1024)
11    capitalizedSentence = sentence.upper()
12    connectionSocket.send(capitalizedSentence)
13    connectionSocket.close()
14
```

The left sidebar shows the project structure with the following folders and files:

- pythonProject
 - tcp
 - client.py
 - server.py
 - udp
 - client.py
 - server.py
 - venv
 - External Libraries
 - Scratches and Consoles

The bottom panel shows the Run output window with the following text:

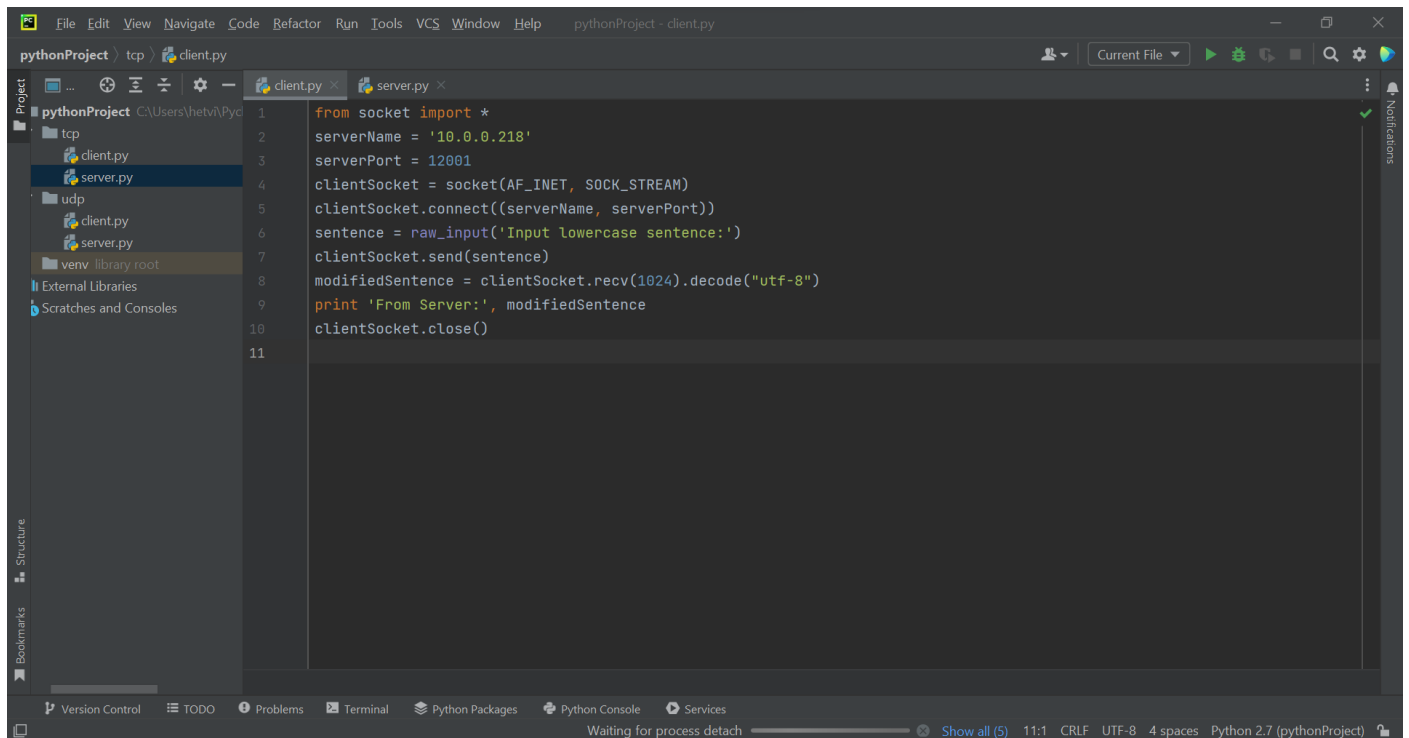
```
Run: server x client x
C:\Python27\python.exe C:\Users\hetvi\PycharmProjects\pythonProject\tcp\server.py
The server is ready to receive
```

The status bar at the bottom indicates the current file is `server.py`, the encoding is `UTF-8`, and the Python version is `Python 2.7`.

This is the output for tcp server where it shows that the server is ready to receive the message from client.

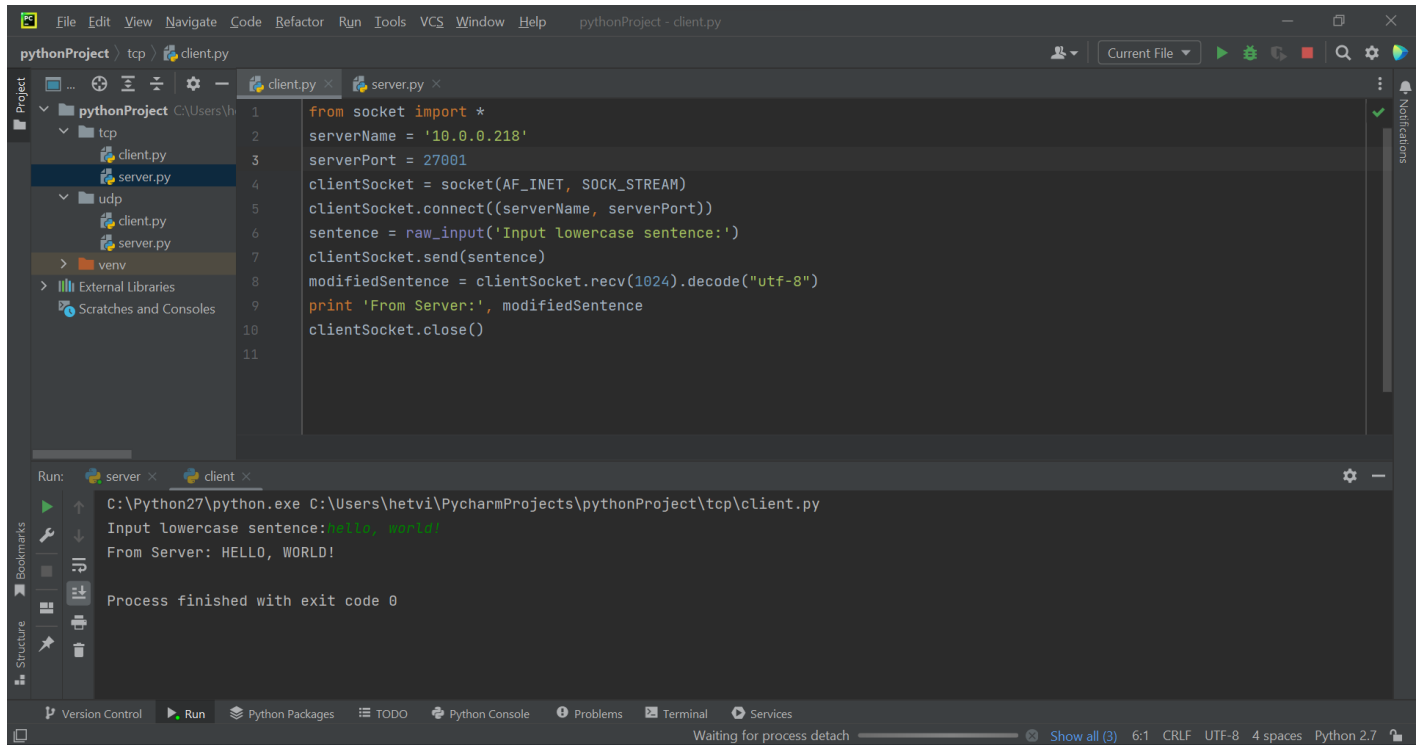
Source code for client:

```
from socket import *
serverName = '10.0.0.218'
serverPort = 27001
clientSocket = socket(AF_INET, SOCK_STREAM)
clientSocket.connect((serverName, serverPort))
sentence = raw_input('Input lowercase sentence:')
clientSocket.send(sentence)
modifiedSentence = clientSocket.recv(1024).decode("utf-8")
print 'From Server:', modifiedSentence
clientSocket.close()
```



This is the source code for tcp client where it asks to input a lowercase sentence which will then output that sentence as an uppercase sentence using tcp server. The tcp client file always uses close() function.

Output for client:



The screenshot displays the PyCharm IDE interface. The main editor window shows the code for `client.py`, which is a Python script for a client socket. The code is as follows:

```
1 from socket import *
2 serverName = '10.0.0.218'
3 serverPort = 27001
4 clientSocket = socket(AF_INET, SOCK_STREAM)
5 clientSocket.connect((serverName, serverPort))
6 sentence = raw_input('Input lowercase sentence:')
7 clientSocket.send(sentence)
8 modifiedSentence = clientSocket.recv(1024).decode("utf-8")
9 print 'From Server:', modifiedSentence
10 clientSocket.close()
11
```

The left sidebar shows the project structure with the following hierarchy:

- pythonProject
 - tcp
 - client.py
 - server.py
 - udp
 - client.py
 - server.py
 - venv
 - External Libraries
 - Scratches and Consoles

The bottom panel shows the Run output for the client process. The output is as follows:

```
Run: server x client x
C:\Python27\python.exe C:\Users\hetvi\PycharmProjects\pythonProject\tcp\client.py
Input lowercase sentence:hello, world!
From Server: HELLO, WORLD!
Process finished with exit code 0
```

The status bar at the bottom indicates the current file is `client.py`, the encoding is `UTF-8`, and the Python version is `Python 2.7`.

This shows us the output for the lowercase input sentence converted to the uppercase sentence.